A Constraint Equation Algebra as a Basis for Haptic Rendering

Matthew Hutchins
CSIRO Mathematical and Information Sciences [6]
GPO Box 664, Canberra, ACT 2601, Australia.
Matthew.Hutchins@cmis.csiro.au

Abstract

Many haptic rendering problems can be expressed in terms of constraints on the motion of a proxy within a virtual environment. This principle is well established for surface rendering, and can also be applied to other types of haptic interaction. A key problem in general constraint based rendering is combining constraints from several sources into a single unified constraint. This paper describes some work in progress toward developing a mathematical framework for manipulating motion constraint equations, and in particular the derivation of a combination algebra for constraints. This work could lead to a system for 6DOF rendering involving non-trivial proxy shapes.

Keywords: haptics, constraints, proxy, 6DOF.

1. Introduction

In a previous PUG paper [4] (and see also [2,3]) we described an approach to haptic rendering based on the use of constraints, and in particular the use of a constrained *proxy*. The use of a proxy (or god-object) for haptic surface rendering is described in [5], and is now a well established technique. The basic idea is that the physical movements of the PHANToM are tracked by a virtual proxy object moving in a virtual environment, as shown in Figure 1. Whereas the PHANToM moves freely in space, the proxy object will encounter virtual objects and fields which will constrain or change its motion. The difference between the free motion of the PHANToM and the constrained motion of the proxy is used as the basis for generating contact forces. This technique is used for surface rendering by preventing the proxy from moving from one side of a surface to the other. It can also be used for other types of haptic interactions, for example: constraining the proxy to a line or plane. A graphical representation of the proxy will usually be rendered in the user interface to provide multimodal feedback. The graphical and haptic properties of the proxy are not necessarily the same — the graphics may be considerably more elaborate than the haptic rendering, for example. In this paper, "proxy" refers to the haptic representation, which may be as simple as a single point.

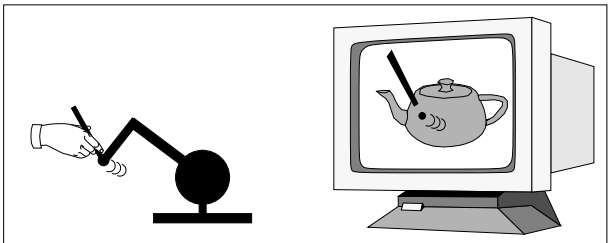

Figure 1: PHANToM and proxy.

The two major haptic rendering SDKs/APIs (Software Development Kits or Application Programming Interfaces) currently available (that we know of) are GHOST from SensAble [8] and Magma from Reachin [7]. Both provide separate facilities for shape based surface rendering and abstract haptic effect or force field rendering. Only the surface rendering interfaces allow manipulation of the proxy position (in GHOST called the Surface Contact Point or SCP). Thus to program a haptic constraint using the proxy technique, one must implement the constraint using the surface interface, or implement a second proxy. The second proxy solution is messy, and doesn't integrate well with surfaces that use the built-in proxy. The surface interfaces are, naturally enough, good for effects that act like surfaces, but awkward to use for more general effects.

The context of the work described in this paper, then, is the development of a new approach to specifying haptic rendering that unifies surface and other types of rendering into a single framework based on constraining the motion of a proxy. This is work in progress, with the majority of the framework still under development. This paper describes some of the mathematical formalisms that have been developed so far for specifying and manipulating constraints on the motion of a proxy.

To provide slightly more context for the mathematics, consider the problem of implementing a haptic scenegraph object, which could be a solid shape, a deformable shape, or some abstract force field. At each traversal of

the scenegraph, the object may be required to solve the problems of collision detection and contact registration. Contact registration means, upon detecting a collision, registering the contact with the rendering system. The contact is the focus of a two-way communication between the scenegraph object and the rendering system. The system must combine the effects of all registered contacts to produce a new proxy position, an output force, and dynamic information to feed back to the scenegraph objects to update their internal state (e.g. deform). One aspect of the contact is the local topology in a neighbourhood of the contact point. This can be expressed as constraints on the motion of the proxy around the contact point. It is essential that the topological information from separate objects, which act independently of each other, can be combined by the system to give a single result. This is the motivation for the algebraic treatment described later in the paper.

2. Motion and constraints

We will assume that the proxy is a rigid body, and its motion is described by rigid body kinematics [1]. At any instant in time, the configuration of a body in space can be described by its position and orientation with respect to some fixed reference "origin". If the configuration of a body in motion is sampled at discrete times, the difference between any two such configurations can be represented[1] by a tuple

$$(A, \hat{\underline{a}}, x, \omega)$$

where A is a point, $\hat{\underline{a}}$ is a unit vector, x is a distance and ω is an angle. Here A is called the anchor point, and together with $\hat{\underline{a}}$ defines a line which is an axis of rotation. The difference in configurations can be interpreted as the effect of a translation of the body along the axis by a distance x, and a rotation around the axis counterclockwise by the angle ω.

We assume that the motion of the PHANToM is an arbitrary continuous motion sampled at discrete times. We wish to approximate this motion over a sampling interval by a simple substitute motion that is easily represented. We choose a form of screw motion where the translation distance and rotation angle change proportionally over time. That is, the difference between two configurations over an interval t will be

$$(A, \hat{\underline{a}}, xt, \omega t)$$

for given constants A, $\hat{\underline{a}}$, x and ω. Thus the complete motion can be represented by a tuple $(A, \hat{\underline{a}}, x, \omega)$ which can be directly computed from the difference between the initial and final configurations of the PHANToM over a sampled interval.

As a body moves in space over time, a point P on the body will move through a curve or trajectory in space which we can describe as $P(t)$. For the proportional screw motion $(A, \hat{\underline{a}}, x, \omega)$, and taking $t \in [0, 1]$, the trajectory of a point is given by the equation

$$\begin{aligned} P(t) \;=\; & P_0 + (-\underline{r}) + xt\hat{\underline{a}} + \underline{r}\cos(\omega t) + \\ & (\hat{\underline{a}} \bullet \underline{r})\hat{\underline{a}}[1 - \cos(\omega t)] + (\hat{\underline{a}} \times \underline{r})\sin(\omega t) \end{aligned} \tag{1}$$

where P_0 is the initial position of P and $\underline{r} = (A - P_0)$. The tangent of the trajectory at any point is given by the derivative of the curve at that point with respect to time:

$$\begin{aligned} P'(t) \;=\; & x\hat{\underline{a}} - \omega\underline{r}\sin(\omega t) + \\ & \omega(\hat{\underline{a}} \bullet \underline{r})\hat{\underline{a}}\sin(\omega t) + \omega(\hat{\underline{a}} \times \underline{r})\cos(\omega t) \end{aligned} \tag{2}$$

This tangent can be thought of as describing the direction that the point P is moving in at time t. Initially, at time $t = 0$, the tangent is

$$P'(0) \;=\; x\hat{\underline{a}} + \omega(\hat{\underline{a}} \times (A - P_0)) \tag{3}$$

We call $x\hat{\underline{a}}$ the translational component of the tangent and $\omega(\hat{\underline{a}} \times (A - P_0))$ the rotational component.

Our goal is to be able to express constraints on the complete motion of the proxy as the combined effect of simple constraints on the motion of individual points in the proxy. One way these constraints would arise is if a point on the surface of the proxy was in contact with the surface of an object in the scene. A constraint on the motion of a point can be expressed as a constraint on the tangent of the trajectory of the point under the motion. For the remainder of this paper we will make two further simplifying assumptions:

- we express motion constraints as constraints on the tangent $P'(0)$ at the start of a sampling interval only, not on the trajectory over the interval;

1. This representation is possible due to a corollary to Euler's theorem that Goldstein [1, p.163] attributes to Chasles.

- we wish to constrain the translational component and the rotational component of the tangent separately.

A more general approach will be left for future work. However, this simplified version may serve as a good enough approximation for the purposes of haptic rendering.

3. Constraint equation algebra

To recap, we have a representation for a particular class of rigid body motions as tuples of the form

$$M = (A, \hat{\underline{a}}, x, \omega)$$

and we have an equation (Eqn. (3)) that describes a tangent of the trajectory of a point under such a motion. By specifying a condition that must be satisfied by the tangent of some point P, we can identify a set of motions that will make the tangent satisfy the condition. Thus a *constraint equation* on the tangent defines a set of legal motions. Clearly, there is a wide variety of conditions that can be placed on the tangent. We wish to choose a useful subset of these conditions and develop a mathematical and computational framework for manipulating them. In other words, we wish to develop a *constraint equation algebra*.

An algebra, in the most general sense as used in algebraic software specification, is simply a collection of sets and functions and relations that satisfy some chosen axioms. To specify an algebra, we need to define what sets there are, what elements are in them, and what operators and relations act on those sets. This is analogous to defining an abstract data type in software. In this case, we wish to define a set of constraint equations, and a single operator to combine pairs of constraint equations. We will want the operator to be idempotent, commutative and associative, so the resulting algebra will have the form of a semi-lattice.

4. The base cases

We start by defining a set of constructors, or generators, or "base cases" for the set of constraint equations. These are the building blocks which will be combined to create the complete set. As we have described, we wish to define these mostly in terms of constraints on the tangent of a particular point at the start of an interval. A useful set of base cases is:

- Free : free motion of the body.
- $\mathsf{PerpTan}(P, \hat{\underline{n}})$: the tangent of P at $t = 0$ is perpendicular to unit vector $\hat{\underline{n}}$.
- $\mathsf{ParaTan}(P, \hat{\underline{n}})$: the tangent of P at $t = 0$ is parallel to unit vector $\hat{\underline{n}}$.
- $\mathsf{FixPoint}(P)$: the tangent of P at $t = 0$ is zero, so that P is fixed.
- Fixed : the entire body is fixed.

We can identify the set of motions permitted by each of these cases by using the tangent equation Eqn. (3). First, some notation for parallel and perpendicular vectors. We define

$$\begin{aligned} \underline{u} \parallel \underline{v} &\equiv (\underline{u} \times \underline{v} = \underline{0}) \\ \underline{u} \perp \underline{v} &\equiv (\underline{u} \bullet \underline{v} = 0) \end{aligned} \tag{4}$$

Then, remembering the assumption that the translation component and the rotation component will be constrained independently, we can derive the following definitions:

$$(A, \hat{\underline{a}}, x, \omega) \in \mathsf{Free} \equiv \mathsf{TRUE} \tag{5}$$

$$\begin{aligned} (A, \hat{\underline{a}}, x, \omega) \in \mathsf{PerpTan}(P, \hat{\underline{n}}) \equiv \\ \{(x = 0)\ \mathsf{OR}\ (\hat{\underline{a}} \perp \hat{\underline{n}})\}\ \mathsf{AND} \\ \{(\omega = 0)\ \mathsf{OR}\ ((\hat{\underline{a}} \times (A - P_0)) \perp \hat{\underline{n}})\} \end{aligned} \tag{6}$$

$$\begin{aligned} (A, \hat{\underline{a}}, x, \omega) \in \mathsf{ParaTan}(P, \hat{\underline{n}}) \equiv \\ \{(x = 0)\ \mathsf{OR}\ (\hat{\underline{a}} \parallel \hat{\underline{n}})\}\ \mathsf{AND} \\ \{(\omega = 0)\ \mathsf{OR}\ ((\hat{\underline{a}} \times (A - P_0)) \parallel \hat{\underline{n}})\} \end{aligned} \tag{7}$$

$$\begin{aligned} (A, \hat{\underline{a}}, x, \omega) \in \mathsf{FixPoint}(P) \equiv \\ \{x = 0\}\ \mathsf{AND} \\ \{(\omega = 0)\ \mathsf{OR}\ (\hat{\underline{a}} \parallel (A - P_0))\} \end{aligned} \tag{8}$$

$$\begin{aligned} (A, \hat{\underline{a}}, x, \omega) \in \mathsf{Fixed} \equiv \\ \{x = 0\}\ \mathsf{AND}\ \{\omega = 0\} \end{aligned} \tag{9}$$

The zero motion that satisfies Fixed will satisfy all of the other constraints. A motion that satisfies $\mathsf{FixPoint}(P)$ must be a pure rotation around an axis through P, and will satisfy any other constraint on P.

5. Composition

We now define a composition operator on constraint equations, denoted $C_1 \oplus C_2$. The motions that satisfy the combination $C_1 \oplus C_2$ should be precisely those that satisfy both of the constraints C_1 and C_2. Thus we define

$$M \in (C_1 \oplus C_2) \equiv (M \in C_1)\ \mathsf{AND}\ (M \in C_2) \tag{10}$$

Another way of saying this is that $C_1 \oplus C_2$ is the intersection of the sets C_1 and C_2. Thus, we know this com-

position operator satisfies the axioms required of a semi-lattice (idempotency, commutativity, associativity), because set intersections do.

The complete set of elements in the algebra is therefore all those generated by the five base cases, plus the composition of any two other elements. This is essentially a recursive definition, and computationally would require a recursive data structure to represent the elements. However, we can make some observations to simplify this. Firstly, suppose that all constraints must be applied to the same point P. This would be the case for a 3DOF rendering system with a single point proxy, or, such as in Magma, a small spherical proxy where all constraints are translated to apply to the centre of the sphere. It turns out that the five base cases completely characterise the system. That is, every combination of two or more constraints applied to the same point are equivalent to a simple constraint applied to the same point. Mostly, the result of $C_1 \oplus C_2$ is either C_1 or C_2 or $\mathsf{FixPoint}(P)$. The only interesting case is

$$\mathsf{NOT}(\hat{\underline{n}}_1 \parallel \hat{\underline{n}}_2) \Rightarrow$$
$$(\mathsf{PerpTan}(P, \hat{\underline{n}}_1) \oplus \mathsf{PerpTan}(P, \hat{\underline{n}}_2) = \mathsf{ParaTan}(P, \hat{\underline{n}}_1 \times \hat{\underline{n}}_2))$$

For the general case where constraints can be applied to different points on the proxy, there are definitely more elements required. However, it appears that the five base cases plus the six pairwise combinations of the non-trivial base cases will be enough to completely characterise the space. So, all constraints could be represented in a flat data structure with eleven types of elements. The proof of this conjecture is work currently in progress.

6. Future work

The combination algebra developed above makes it possible to reduce a set of independent constraints to a single constraint that must be satisfied by the motion of the proxy. Given a potential motion and a constraint, there is a straightforward decision procedure to determine if the motion satisfies the constraint. However, if a motion of the proxy does not satisfy the constraint, it is necessary to find an alternative motion that does. This is always possible (the zero motion satisfies any constraint), so in fact the problem is to find the "best" alternative motion. The optimal solution may be different for each of the eleven constraint cases (of course, the Fixed and Free cases are easy!) Once the development of the algebra is complete, this will be the next problem to be solved. Some cases have been solved already. We assume that, at present, most haptic rendering will be done using a 3DOF output device, or a 6DOF device where the rotational fidelity is less than the translational fidelity. Therefore it is best to try to match the rotation component of the motion as closely as possible. This will minimise the discrepancy between the orientation of the haptic device and the orientation of the proxy, which will minimise the required torques. Note that the screw motion representation naturally isolates the translation and rotation components of the input motion.

As described in the introduction, this formulation of constraint equation algebra is just one part of a larger specification for a new approach to haptic rendering. Our future work will be the continued development and implementation of this approach.

References

[1] Herbert Goldstein. Classical Mechanics. Second edition, Addison-Wesley, 1980. Chapter 4, pages 128–187.

[2] Chris Gunn and Paul Marando. Haptic constraints: guiding the user. In proceedings of SimTecT '99, Melbourne, Australia, March 1999. Pages 261–264.

[3] Matthew Hutchins. Software components for haptic constraints. In proceedings of the SPIE Vol. 3957, Stereoscopic Displays and Virtual Reality Systems VII, 2000. Pages 423–432.

[4] Matthew Hutchins and Chris Gunn. A haptic constraints class library. In Salisbury, J.K. and Srinivasan, M.A. (Eds), Proceedings of the Fourth PHANTOM Users Group Workshop, AI Lab Technical Report No. 1675 and RLE Technical Report No. 633, MIT, November 1999.

[5] C.B. Zilles and J.K. Salisbury. A constraint-based god-object method for haptic display. In proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems, August 1995. Pages 146–151.

Web Cites

[6] CSIRO Mathematical and Information Sciences. http://www.cmis.csiro.au

[7] ReachIn Technologies. http://www.reachin.se

[8] SensAble Technologies. http://www.sensable.com